CRT-2001-08

Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation

ADAPTIVE MEMORY PROGRAMMING FOR A CLASS OF DEMAND RESPONSIVE TRANSIT SYSTEMS

by

Federico Malucelli
Maddalena Nonato
Teodor Gabriel Crainic
François Guertin

Mars 2001

102-02898/

PUBLICATION
CRT-2001-08

*ADAPTIVE MEMORY PROGRAMMING FOR A CLASS OF DEMAND RESPONSIVE TRANSIT SYSTEMS**

by

Federico Malucelli[1]
Maddalena Nonato[2]
Teodor Gabriel Crainic[3]
François Guertin[4]

* This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Fonds pour la Formation de Chercheurs et l'Aide à la Recherche (FCAR), the Quebec-Italy cooperation program of the Ministry for Foreign Affairs of Italy and Ministry for International Relations of the Province of Quebec.

1 DEI – Politecnico di Milano malucell@elet.polimi.it
2 DIEI – Università di Perugia nonato@istel.ing.unipg.it
3 Département de management et technologie, Université du Québec à Montréal, et Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 theo@crt.umontreal.ca
4 Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 guertin@crt.umontreal.ca

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2001

Résumé

Nous examinons un système de transport public adaptatif (DAS), défini comme un système hybride intégrant des lignes à service traditionnel et des lignes qui s'ajustent à la demande. Les lignes de type (DAS) servent selon un horaire pré-établi un ensemble d'arrêts obligatoires. Des arrêts optionnels peuvent être activés entre deux arrêts obligatoires afin de répondre à la demande. Dans ce contexte, les routes et les horaires des véhicules doivent être modifiés afin de satisfaire le plus grand nombre de requêtes possible, tout en respectant les contraintes de fenêtre-horaire aux arrêts obligatoires.

Nous présentons une description générale d'un tel système, nous examinons les différences avec les systèmes classiques de transport à la demande, nous discutons les applications potentielles et les méthodes pour sa planification et opération. En effet, la structure mathématique particulière de systèmes de type (DAS) exige, même dans les cas les plus simples, des méthodes de résolution innovatrices. Nous proposons une méthode méta-heuristique efficace pour le cas simple d'une ligne et d'un passage unique. Tirant avantage de la structure du problème, la méthode combine des outils sophistiqués de programmation mathématique et de recherche avec tabous. Les résultats expérimentaux montrent que le modèle simple peut être résolu efficacement et, conséquemment, qu'il peut véritablement être considéré comme un élément constitutif important de stratégies globales pour le problème général.

Mots-clés : transport collectif, systèmes adaptatifs, méthodes de recherche avec tabous.

Abstract

In this paper we consider a Demand Adaptive System (DAS) which is intended as a hybrid public transportation system that integrates traditional bus transportation and on demand service. DAS lines regularly serve a given set of compulsory stops according to a predefined schedule and regardless of current demand. Between a compulsory stop and the next, optional stops can be activated on demand. Vehicles have to be rerouted and scheduled in order to satisfy as many requests as possible, complying with passage-time constraints at compulsory stops.

This paper provides a general description of DAS, and discusses potential applications and solution methods, emphasizing differences and analogies with classical Demand Responsive Systems. The particular mathematical structure of DAS requires innovative solution methods even when addressing its simplest version, the single vehicle, single line case. An efficient meta-heuristic algorithm based on adaptive memory ideas has been developed for this case. The method integrates sophisticated mathematical programming tools into a Tabu Search structure, taking advantage of the particular structure of the problem. The methodology is briefly discussed and experimental results are presented for the single line case. We show that the basic case can be efficiently solved, thus providing efficient algorithmic building blocks for more comprehensive approaches tackling the general case.

Key words : Public Transportation, Demand Adaptive Systems, Tabu Search.

1 Introduction

Traditional public transport is evolving towards more flexible services and operation modes in order to capture additional demand and increase profitability. Several flexible services that target people with particular needs are regularly operated in several urban centers. *Dial-a Ride (taxi-bus) services* and *on-demand door-to-door transportation* for handicapped and elderly people are examples of such *Demand Responsive Systems* (DRS) [8, 1]. Yet, the needs of several groups of customers are not addressed by these services. This is particularly the case when demand is relatively low, such as during evenings and night time, in suburban and rural areas.

To capture this additional demand, the service quality of regular transit lines should be significantly increased. However, this goal appears difficult to attain if the transit company must operate within reasonable economic limits. One approach increasingly explored by transit authorities is to introduce some flexibility into the regular line operations such that customers may be directly reached, while still controlling the operation costs. To provide personalized services on such a large scale with reasonable investments, new service models have to be designed, which exploit the integration of on-demand services with the existing transportation network, and improve the accessibility and efficiency of the whole system. Intelligent Transportation Systems offer the technological context for the deployment and operation of such systems through advanced location, communication, guidance, and information management systems. To render such user-oriented, flexible services efficient and profitable, one has also to develop new optimization models and effective algorithms that are the enabling factor of comprehensive management and control tools.

The contribution of this paper is to introduce *Demand Adaptive Systems* (DAS), a new type of public transportation service that integrates traditional bus transportation and personalized on-demand services, discuss analogies and differences of DAS and traditional (DRS), propose potential application frameworks, and show how DAS can be efficiently managed by ad hoc solution methods despite the complexity of the associated mathematical representation.

The paper is organized as follows. The proposed service model is formally described in its most general case in Section 2. We then focus on the particular case of the efficient operation of one vehicle making a tour of one line. The mixed integer programming formulation of the related optimization problem is provided in Section 3. As shown in [5], this problem constitutes the basic element that makes up the more general problem and, thus, efficient solution methods for this case are essential. The computation of upper bounds is discussed in Section 4 , while Section 5 is dedicated to the description of the heuristic solution approach and its main ingredients.

Computational results are discussed in Section 6. Finally, conclusions and further directions of work are discussed in Section 7.

2 Demand Adaptive Systems

The transportation system discussed in this paper, denoted *Demand Adaptive System* (DAS), is a hybrid system that combines the regularity of fixed-line transportation and the flexibility of on-demand service. A multiple line DAS is designed as follows. Each trip of the line is described, as usually, in terms of a set of time-tabled trips. The stops in the original timetable are called *compulsory stops*. Such points are locations where a demand for transportation may be naturally assumed to be present, such as, for example, the stops of an express line, stops at rail or subway stations, or any other point that must be serviced according to the policies of the local transit authority. To introduce some flexibility into the vehicle routes, the vehicle is allowed to transit by each compulsory stop during a *time window*. Beside compulsory stops, a given set of stops to be activated on demand (*optional stops* hereafter) is available to the users. Each optional stop is assigned in a unique way to a pair of compulsory stops which are consecutive along the line and it can be serviced by the vehicle on its way from a compulsory stop to the next. Each line may be operated by several vehicles, starting from the line terminal according to a defined timetable.

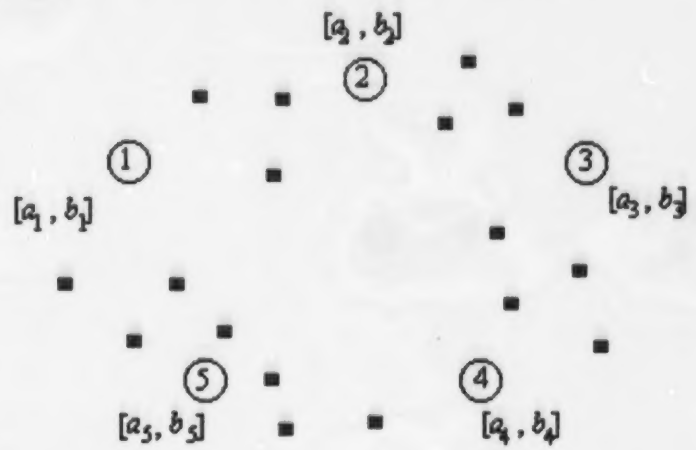

Figure 1: The compulsory and optional stops of a DAS line

In Figure 1, optional and compulsory stops of a circular DAS line are depicted. Compulsory stops are progressively numbered according to the visiting order, and the

associated time windows are shown. Black squares denote optional stops.

From the demand point of view, one can distinguish two classes of users. So called *passive users* make use of the transportation service in a traditional way, i.e. boarding and alighting at compulsory stops. No reservation is necessary since vehicles are guaranteed to serve each compulsory stops within a given time windows. Users of the second class, so called *active users*, ask for a ride boarding or alighting at an optional stop. Active users must issue a *service request* and specify pick up and drop off stops, as well as earlier departure and latest arrival times.

In the absence of requests involving optional stops, the vehicle travels along the shortest path on the road network between each pair of consecutive compulsory stops. Accepting a request implies rerouting the vehicle to serve the pick up and the drop off stops within the required time windows. The vehicle detour may impact on the transit time at the following stops. In Figure 2, a possible itinerary of the vehicle on the line of 1 is shown; all optional nodes visited along the itinerary correspond to boarding or alighting stops of requests; requests may have either both stops located at optional nodes (as s_1, d_1 for request r_1 and s_2, d_2 for request r_2), or the boarding stop only, as for r_4 where d_4 is compulsory, or the alighting stop only as for r_3 where s_3 is compulsory.

Travelling times on the arcs of the physical network are assumed known. Transfers between different lines, at line intersection points, are allowed at compulsory stops only, where intersection with fixed lines may also occur. If a request involves different lines, service time must allow for line transfer. Note that a request may be served by several possible itineraries, in particular when the flexible system is integrated with the fixed line system.

Several DAS models can be devised, depending on the policy used to deal with requests. In this paper, we focus on a specific case according to which requests are rejected if their acceptance causes the tour to become infeasible; a request rejection corresponds to the offer of a personalized service at a more expensive fare. On the other hand, once a request has been accepted, the user must be boarded and alighted exactly at the desired stops and within the requested time constraints.

In the basic model, namely DAS1 [13], a single line circuit is traversed once by a single vehicle and the most profitable subset of requests that can be feasibly served must be selected. This model may be readily generalized to multiple intersecting lines with several vehicles traversing each line; here, users may travel from stops of a line to stops of another line on vehicles connecting at compulsory stops. The solution strategies developed for DAS1 provide the building blocks of more comprehensive formulations.

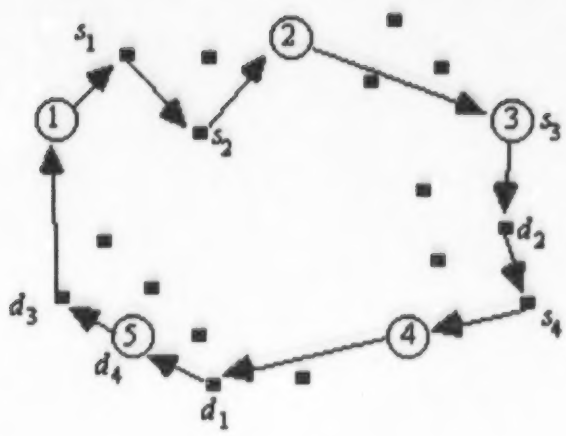

Figure 2: A set of requests and a vehicle tour on a DAS line

2.1 Demand Adaptive Systems versus Dial-a-ride

Demand Adaptive Systems are intended to play quite a different role in the family of Demand Responsive Systems from that of classical Dial-a-Ride systems (DAR). Differences arise not only relative to the service purpose, but also involve the mathematical structure of the problems and the associated solution methods. We address these aspects in the following.

In DAS as in DAR, customers are allowed to request service between two given points and thus induce a detour in the vehicle route. However, in DAS, the transportation service is always guaranteed, without any booking, at the compulsory stops of the line. Thus, a basic service is always assured to *passive users*.

Regarding intended users, DAR was originally introduced to serve people with special needs or, in general, as a highly subsidized service with precise targets. Once that broader classes of users have been entitled to the service, transit companies face the difficult challenge of how to cut down costs while not losing the acquired ridership. DAS, on the other hand, is not devoted to any special class of users in particular. Indeed, it represents a smooth way for transit agencies to migrate towards new transportation organizations, where users are expected to interact with the company call center. In fact, while *active* users personalize their itineraries by issuing requests, DAS still provides transportation without booking to *passive* users at compulsory stops. This may help users get acquainted progressively with the new kind of service.

From the infrastructure and technological aspects point of view, it is known how

much flexible services take advantage of precise vehicle location systems as well as of information systems that provide users with reliable information about times, itineraries, alternatives, etc. These issues are particularly central when the management of the requests is on-line and in real time. As far as the vehicle location system is concerned, which is usually a critical point especially in urban settings, note that in DAS, vehicle's position can be determined with a good approximation and without any expensive equipment, by exploiting the structure of the service. Since vehicles follow a basic itinerary, they can be located by knowing the last visited compulsory stop. The same information cannot be retrieved so easily in the DAR case because vehicles move with no tights in the service area, so that more sophisticated equipment must be utilized.

Regarding potential applications of DAS, one of the most suitable is given by a flexible version of the classical *feeder bus* [4]. Feeder buses connect low demand areas to high capacity swift transport lines, either underground or light rail. In the flexible version, main line connections would occur at the compulsory stops of the flexible line and users could access the swift line by riding the feeder bus from an optional stop. Moreover, DAS lines could be operated to provide connection among multiple high capacity rapid lines. In this context, user itineraries would be composed of portions of trips operated by the flexible system and trips riding the regular, fixed line transit system. Furthermore, out of peak hours, several fixed lines operating in close by areas could be replaced by a single DAS line that would "virtually" cover the whole area, while guaranteeing regular connection at particular sites. In general, DAS is a system that may address very well the needs of relatively low demand areas (e.g., North American city suburbs) or periods (e.g., non-rush hours, evenings, week ends, etc.) or, broadly speaking, all those cases where a basic transportation service must be guaranteed even though a traditional type of transit system is not economically worthy and efficiency becomes a critical issue. In this context, it is clear how much the integration of the flexible and traditional transportation systems is important and how both systems can take advantage from being operated as a unique entity.

Significant differences between DAS and classical dial-a-ride systems also arise in the mathematical structure of the optimization problems.

In the optimization problem associated with DAS (Section 3), time windows are present only at compulsory stops and the stop set is partitioned into subsets which must be visited in a predefined sequence. Therefore, the problem does not consist in partitioning and sequencing the nodes of a graph into disjoint routes that fulfill time windows and vehicle capacity constraints as in the DAR case. Rather, the problem requires to select which nodes have to be visited in each line trip, and the route definition can be decomposed into smaller subproblems, one for each portion of trip between two consecutive compulsory stops. In DAS, a single node is potentially asso-

ciated to several requests, while in DAR each node is either a boarding or an alighting node of a single request. DAS is less constrained in the way nodes are sequenced; indeed, the feasible region encompasses any time feasible itinerary. On the other hand, DAS requires the solution of a selection problem of pairs of nodes.

DAR problems are usually addressed with vehicle routing algorithms where the sequencing of nodes plays an important role. Due to the significant differences in the modelling and mathematical structure between the optimization problems related to DAS and DAR, a straightforward application of vehicle routing algorithms does not appear a viable approach for tackling DAS even in its simplest version. The challenge then is to devise new approaches that, based on innovative as well as classical methodologies, exploit the problem structure. In this paper, we concentrate on the basic problem, since efficient and effective algorithms for solving DAS1 constitute valid and essential tools to tackle the more general case. In fact, the same solution strategies developed for DAS1 may be used as building blocks for more comprehensive formulations. For example, the high parallelization potential of the general problem could be exploited by an iterative approach where, at each step, the subproblems related to each single line trip are solved in parallel by way of the specific methods developed for DAS1.

3 Mathematical Formulations for DAS

Let us introduce some notation to formally state problem DAS1. Consider a line structured as a circuit, served by a single vehicle ending its tour at the starting terminal. Along the tour, the vehicle traverses a sequence $H = \{f_1, f_2, \ldots, f_{n+1}\}$ of $n+1$ compulsory stops according to a given order, with $f_{n+1} = f_1$. For each stop f_h a time window $[a_h, b_h]$ is defined, $h = 1, \ldots, n$; the vehicle must leave f_h not earlier than a_h and not later than b_h, but may arrive before a_h; b_{n+1} is the maximum tour completion time and $a_1 = b_1$ the starting time from the terminal.

A set F_h of optional stops is associated to each pair of consecutive compulsory stops $\langle f_h, f_{h+1} \rangle$. The vehicle passes by an optional stop only if a related boarding or alighting request at that stop has been issued. Sets F_h are mutually disjoint. Given any pair $\langle f_h, f_{h+1} \rangle$, we can define a directed graph $G_h = (N_h, A_h)$, such that $N_h = F_h \cup \{f_h, f_{h+1}\}$ is the stop set and $A_h \subseteq N_h \times N_h$ is the set of arcs connecting the stops. G_h will be referred to as *segment* h. Finally, $G = (N, A)$ is the whole graph, where $G = \bigcup_h G_h$. Travelling times and costs of the physical network are given. Travel time τ_{ij} and travel cost c_{ij} for each $(i, j) \in A$ of consecutive stops are strictly positive, with τ_{ij} representing the duration of the shortest path from i to j including the stopping time at i. Then the subgraph induced by F_h is complete. We assume

that triangular inequalities hold but time and cost matrices may be asymmetric.

In the following, we focus on a path formulation that offers a good insight in the problem nature and it is a natural support of efficient solution algorithms. An equivalent arc formulation is discussed in [12]. Let P_h be the set of paths in G_h from f_h to f_{h+1}. The vehicle itinerary in segment h is a path $p \in P_h$ with travel time $\tau(p)$ and cost $c(p)$ equal to the sum of travel times and costs of its arcs, respectively:

$$\tau(p) = \sum_{(i,j)\in p} \tau_{ij}; \qquad c(p) = \sum_{(i,j)\in p} c_{ij}. \tag{1}$$

Let t_h be the starting time from f_h; w.l.g. we assume $t_1 = a_1$. A sequence of n paths, selected one from each segment, forms a *tour* q starting and ending at the terminal. Let $p_h(q) \in P_h$ be the path chosen in segment h. The arrival time at the end of the segment, i.e. at stop f_{h+1}, is then $t_h + \tau(p_h(q))$. Tour q is *time-feasible* if

(i) $t_h + \tau(p_h(q)) \le t_{h+1} \qquad h = 1, \ldots, n-1;$

(ii) $t_n + \tau(p_n(q)) \le b_{n+1};$

(iii) $a_h \le t_h \le b_h \qquad h = 2, \ldots, n.$

Then P_h is restricted to the set of all elementary paths from f_h to f_{h+1} with travel time less than or equal to $b_{h+1} - a_h$ for all segments $h = 1, \ldots, n$.

Let R be the *request set*, where request $r \in R$ is defined as a pair $\langle s(r), d(r)\rangle$ of boarding and alighting stops. We say that $s(r)$ is the *mate* of $d(r)$ with respect to request r, and viceversa. We assume that, for each request boarding and alighting at optional stops, the corresponding nodes do not belong to the same segment. Such an assumption is quite realistic, since any two optional stops within the same segment are relatively close to each other. Because of this assumption, no precedence constraints must be enforced between stops within the same segment, while precedence constraints regarding the boarding and alighting stops of each request are implicitly handled by the sequencing of compulsory stops. A *benefit* $u(r) > 0$ is associated with each request $r \in R$. It may represent the ticket fare or the value of a more complex utility function which can take into account also level of service aspects.

Let $Q = \{q\}$ be the feasible tour set. Let $N(p_h(q))$, $h = 1, \ldots, n$, be the node set of $p_h(q)$, i.e. the *stops served* in segment h, and let $N(q) = \bigcup_h N(p_h(q))$ be the set of stops served in q. Then, $R(q) \subseteq R$, the subset of requests satisfied by tour $q \in Q$ is given by:

$$R(q) = \{r \in R : s(r), d(r) \in N(q)\};$$

$u(q)$, the *benefit* of tour q, is given by:

$$u(q) = \sum_{r \in R(q)} u(r),$$

while the *global cost* $c(q)$ is the sum of the costs of the paths in q. DAS1 consists of finding a tour $q^* \in Q$ of maximum profit, i.e. maximizing the difference between benefit and cost:

$$u(q^*) - c(q^*) = \max\{u(q) - c(q) : q \in Q\}.$$

A path-based mathematical model can be given by introducing the following variables:

- z_p^h, $\forall p \in P_h, h = 1, \ldots, n$;
- y_r, $\forall r \in R$;
- t_h, $h = 1, \ldots, n$;

where $z_p^h = 1$ if path $p \in P_h$ is chosen, $z_p^h = 0$ otherwise, $y_r = 1$ if request r is satisfied, $y_r = 0$ otherwise, and t_h is the starting time from f_h, and $t_1 = a_1$.

In the example depicted in Figure 3, boarding and alighting stops of four requests are shown; variables z_0^h are associated to the direct path from f_h to f_{h+1} serving no optional stop, which are depicted as a straight line. An alternative path is provided, associated to variables z_1^h and depicted with dotted lines, for $h = 1, 2, 3, 5$. If variables z_1^1, z_1^2, z_1^3,z_0^4, z_1^5 are equal to 1, then all requests are satisfied; if $z_1^1 = 0$ or $z_1^3 = 0$, request r_1 is dropped; if $z_1^5 = 1$ request r_3 is satisfied.

$$(das1p): \; Z(y,z) = \max \sum_{r \in R} u(r) y_r - \sum_{h=1}^{n} \sum_{p \in P_h} c(p) z_p^h$$

$$y_r \leq \sum_{p \in P_h} \delta_{s(r),p} z_p^h \qquad h = 1, \ldots, n, \forall r : s(r) \in N_h \tag{2}$$

$$y_r \leq \sum_{p \in P_h} \delta_{d(r),p} z_p^h \qquad h = 1, \ldots, n, \forall r : d(r) \in N_h \tag{3}$$

$$\sum_{p \in P_h} z_p^h = 1 \qquad h = 1, \ldots, n \tag{4}$$

$$t_h + \sum_{p \in P_h} \tau(p) z_p^h \leq t_{h+1} \qquad h = 1, \ldots, n-1 \tag{5}$$

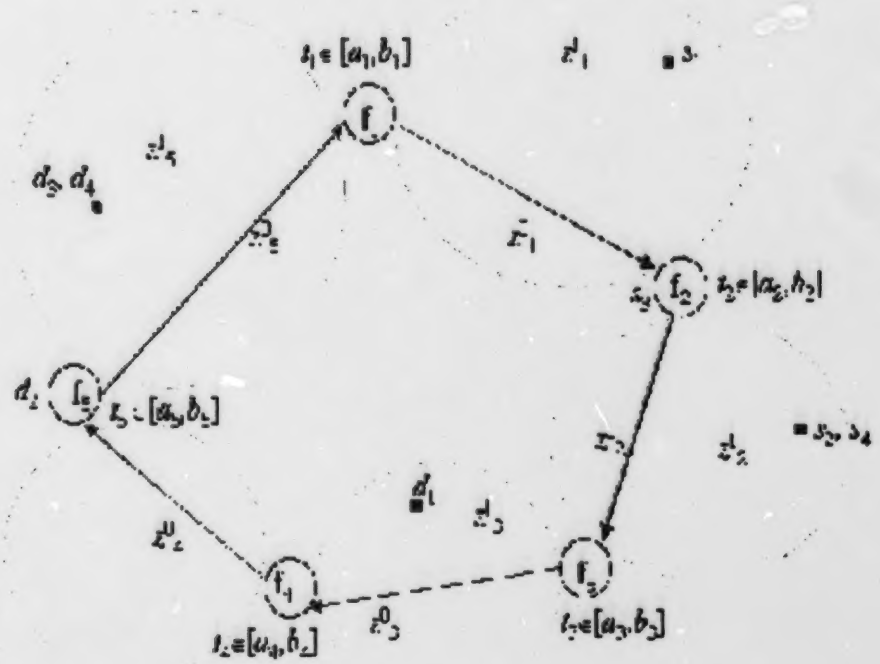

Figure 3: an example of paths and requests

$$t_n + \sum_{p \in P_n} \tau(p) z_p^n \leq b_{n+1} \tag{6}$$

$$a_h \leq t_h \leq b_h \qquad h = 1, \ldots, n \tag{7}$$

$$y_r \in \{0,1\} \qquad \forall r \in R$$

$$z_p^h \in \{0,1\} \qquad \forall p \in P_h, h = 1, \ldots, n$$

where $\delta_{s(r),p}$ is a constant equal to 1 if the boarding stop of the request is a node of path p, and 0 otherwise; similarly, $\delta_{d(r),p} = 1$ if $d(r) \in N(p)$, and 0 otherwise. A request r is served if and only if the tour visits both stops $s(r)$ and $d(r)$. We say then that the tour is *request-feasible* with respect to r. Constraints (2) and (3) link the choice of the path to the served requests, coupling boarding and alighting stops for each request. Constraints (4) impose the selection of one path for each segment, while constraints (5), (6), and (7) concern time-feasibility. Problem ($DAS1P$) is NP-Hard since the travelling salesman problem reduces to a particular instance.

Note that in the present formulation capacity issue is not addressed. Indeed, in DAS framework, passive users board the vehicle at compulsory stops without prior system knowledge; therefore, capacity can be dealt with for active users only. Even thought the system is designed to operate in a low demand setting, where capacity should not be a tight constraint, the model and the proposed solution approach could be generalized to handle active users related capacity constraints.

Commercial Mixed Integer Programming solvers fail to solve even reasonable size instances, as shown in Section 6. Even the computation of an upper bound is not a trivial issue: the linear relaxation of any arc-based Integer Linear Programming model provides quite a loose bound. Heuristics are thus necessary to cope with practical problems. Yet, classical vehicle routing heuristics, constructive or improving (e.g., [15], [9], [10], [14], etc.) much used for dial-a-ride problems, are not appropriate either. Indeed, such methods are based on arc and node manipulation, (principally insertion, deletion, and exchange of stops in and out of the current tour); the effect of these operations on the solution value cannot be uniquely determined in DAS1 framework, since it depends on the solution at hand. Moreover, DAS1 requires the solution of a selection problem which involves pairs of nodes in different segments. This suggests the development of alternative methodologies and motivates the search for specific tools that exploit the particular structure of the problem. The structure of the formulation suggests to take advantage of path-based decomposition and work on paths, rather than on single nodes, and search good solutions by swapping paths in and out of the current tour. We present such a methodology in Section 5.

4 Upper bounds

In this section, we briefly describe how to compute an upper bound to (*das1p*). We also emphasize how the information provided as a by-product by one of the bounding procedures can be used in the heuristic solution algorithms described in the next section.

We considered three possible ways to limit from above the value of the optimal solution of (*das1p*).

The first approach considers the LP-relaxation of (*das1p*). This problem has potentially an exponential number of design variables z_p^h. Therefore, even the solution of the continuous relaxation is a difficult task. This kind of problems are usually tackled by column generation. Here, the formulation of the problem is not completely given, as an initial subset of columns is considered while more are added as needed. The generation of columns (corresponding to variables z_p^h) with positive reduced costs

(i.e. potentially useful for improving the solution) requires the computation of an elementary path p for each segment h with maximum length and travel time no longer $b_{h+1} - a_h$ in a suitable graph. Lengths are given by reduced costs computed with respect to the dual variables of the LP relaxation of ($das1p$) defined on the subset of columns at hand.

A second approach considers alternative arc-based formulations characterized by a polynomial number of variables [12]. In such formulations it is necessary to introduce an additional time variable for each node in order to avoid cycles, as discussed in a more general setting in [7]. As a side effect, the linear relaxation of this formulation is by far looser than the linear relaxation of model ($das1p$). Nevertheless, using these models, it is easier to strengthen the formulation by introducing valid inequalities.

A third approach to compute an upper bound to DAS1 is the Lagrangean relaxation of model ($das1p$), as summarized next. We apply a Lagrangean relaxation to constraints (5) with multipliers μ, and Lagrangean decomposition to variables y_r with multipliers λ. This yields a set of n separable subproblems, $(P(h)_{\lambda,\mu})$, $h = 1, \ldots, n$, which, although still NP-Hard, are rather tractable. In fact, each $(P(h)_{\lambda,\mu})$ reduces to the search of a maximum weight path with a constraint on maximum travel time. Positive cycles may occur because of weight definitions, but, due to time windows constraints, the problem is bounded and can be solved by using the techniques described in [7]. Further details on the efficient computation of an upper bound to DAS1 are provided in [12].

Since the solution of $(P(h)_{\lambda,\mu})$ for any set of multipiiers λ and μ yields a path from h to $h+1$, each evaluation of the Lagrangean function yields a tour, generally not time-feasible, given by the set of the n paths associated with the solution of the n subproblems, one for each segment. The Lagrangian Dual, i.e. the problem of minimizing the Lagrangean function, is solved by way of a bundle algorithm [2] which provides a vector of optimal multipliers, a collection of feasible paths for each segment, and an upper bound to the problem. The collection of these paths, called *promising paths* hereafter, provides a good starting base for heuristic methods that assembles feasible tours by selecting one path for each segment. Such *path-based meta-heuristics* are described in the next section.

5 A Meta-heuristic Solution Procedure

We present a meta-heuristic that integrates into a Tabu Search framework the Lagrangean Relaxation procedure and a Memory-Enhanced Multi-trial Randomized Greedy heuristic. Each individual procedure takes advantage of the particular struc-

ture of the problem, and has been extensively analyzed both as a stand-alone solution method and in a comparative framework [3]. In this paper we present the procedure that gave the overall best performance.

The meta-heuristic is built around a local search phase that iteratively attempts to improve a feasible tour. The core of the algorithm consists of swapping paths in and out of a tour. That is, starting from a feasible solution, the current tour is modified at each step by selecting one path not in the tour from a given set of promising paths and swapping it with the path currently in the tour, in the corresponding segment. The set of promising paths generated by the Lagrangean Relaxation provides the initial pool of candidate paths. This set is dynamically updated by modifying, deleting, and adding new paths to the set in order to keep track of the evolution of the algorithm and, possibly, capture more profitable solutions. Two mechanisms are used to further enhance the search. On the one hand, the search is intensified around the most promising solutions found during the local search phase. On the other hand, a constructive greedy heuristic procedure which exploits the memory of the already generated solutions is used to diversify the search.

More formally, consider a pool of paths $P' = \bigcup_h P'_h$, $P'_h \subseteq P_h$, where P'_h corresponds to the set of available paths for segment h. Let q be the current time-feasible solution. At each step, the *best* path in P' with respect to q for a given segment h is selected according to an evaluation function – the *score* – which depends on the particular solution strategy and implementation. A new tour may then be built from q by *swapping* the selected path with $p_h(q)$, if the new tour is still time-feasible; the next best path is selected otherwise. The path swapping operation thus represents the core move of the path-based metaheuristics we introduce. Then, the neighborhood of the current solution q is the set of all time feasible tours which can be obtained by way of a single path swap move, while the pool P' defines the restricted neighborhood on which swaps are actually evaluated and performed.

The proposed metaheuristic (referred as hybrid in the extensive study [3]) proceeds according to three steps:

1. **A Neighborhood Search** (the "local" search phase) is performed until a given number of unimproving consecutive iterations are observed. At each iteration, a segment is first selected. The non-tabu path in the pool that ensures the best swap with the path currently in the segment is then identified, the move is implemented, and the memories and best solutions are updated. A local improvement of the current tour is performed by reordering the nodes of the selected paths and by inserting new nodes according to a specific priority function (mending step).

2. If the local best solution has been improved during the last Neighborhood Search phase, an **Intensification** phase is performed to attempt to further improve it. The basic principle is to "lock" in the solution the elements that appeared most often in the best local solution. However, building this phase on paths may rapidly lead to situations where no paths may be moved out of or in the solution. We therefore work with requests: we forbid paths that do not contain requests that have appeared most often in the best local solution to enter the tour for a certain *tenure*. A new Neighborhood Search phase is started following Intensification.

3. If the local best solution has not been improved during the last Neighborhood Search phase, we **Diversify** the search. In this phase, a feasible tour is constructed from scratch that is by performing n successful swap moves starting from the basic tour $\bar{q}$ as the initial solution. $\bar{q}$ is composed by the n basic paths $\bar{p}_h$, $h = 1, \ldots, n$, defined as the minimum travel time path from f_h to f_{h+1} without intermediate stops. Each successful swap is accomplished by a sequence of trials. First, the current *working* segment is randomly selected for swapping purpose among those not yet inspected. At each trial, the best candidate path not yet inspected, if any, is searched for among all paths of the working segment currently in the pool. The selected path is marked as inspected, and is tentatively swapped in the current tour. Either the new tour is time-feasible, or the trial is repeated. If all paths of the working segment in the pool have already been inspected, the corresponding basic path will belong to the resulting solution (*null swap*), leading to the rejection of all requests involving an optional stop of that segment. If the new tour is time-feasible, it is improved by deleting profitless nodes, that is those nodes related to requests whose mate node has not been included in the tour; the pool is updated by inserting the new paths generated during the search process, and a new working segment is selected.

   A new Neighborhood Search phase is initiated from the resulting tour.

Several **memories** are defined to guide the search and to evaluate moves. All memories are shared by the Tabu and Greedy heuristics.

- A *short term tabu* memory. Implemented as a tag chosen randomly in a prespecified interval, it indicates the iteration a path may be considered again for inclusion in the tour.

- A path *frequency* memory: for how many iterations a path has been in the best global tour.

- A request *frequency* memory in the current Neighborhood Search phase: for how many iterations a request has been served by the best local tour.

- *Promising* and *realized request memories*. The contribution of node i to the final solution depends on the selection of its mates. For each r such that $i = s(r)$ or $i = d(r)$ and such that the mate of i with respect to r has neither been chosen nor rejected in the current iteration, what matters is the probability of satisfying r once that i has been selected. Such a probability can be estimated by keeping a record of favorable cases (number of times the selection of i has been followed by the selection of its mate) over possible cases (number of times i has been selected before its mate was examined). To this purpose, for each request r, we record the number of times node $s(r)$ has been selected before $d(r)$ was processed, and the number of times the selection of $d(r)$ actually followed, thus satisfying r (the same applies conversely for the boarding node of each request). Let us call these *promising* and *realized* (*held*) *memories* $s_{pr}(r)$ and $s_{hd}(r)$, and $d_{pr}(r)$ and $d_{hd}(r)$, respectively. These counters are updated at the end of each iteration and are long term memories.

To compute the **score** of a potential move, the sum of the differences in the profits associated to the nodes of the paths and the associated transportation costs is calculated to obtain the *exact* move evaluation. However, this evaluation does not take into account any increasing benefit that may come from including extra nodes into the current tour. Moreover, in the constructive phase, we work on an incomplete solution, so that an exact evaluation is not possible. Therefore, we try to approximate the performance of each path on the basis of solutions visited previously during the search, using potential contribution of requests at nodes.

To evaluate the potential contribution of a node to the final solution we make use of the above mentioned promising and realized memories, as follows. Call $\tilde{P}s(r)$ and $\tilde{P}d(r)$ the approximations of the probability of holding the boarding and alighting nodes, respectively, of request r. Then, $npr(i)$, the potential benefit of a node i, is computed as the sum of the revenue of r on all requests r such that i is either $s(r)$ or $d(r)$, times the approximation of the above mentioned probability.

$$npr(i) = \sum_{r:i=s(r)} u(r)\tilde{P}s(r) + \sum_{r:i=d(r)} u(r)\tilde{P}d(r). \tag{8}$$

Such probabilities are estimated from the ratio of favorable cases over possible cases, whose record is kept in the promising and realized request memories.

In the local search phase, the node potential evaluation of a move is computed as the sum of the exact evaluation and the potential revenue of the not yet satisfied requests associated to the nodes of the incoming path. In the constructive phase, is made of two components: for the segments for which a path has already been selected,

the exact benefits are computed. For the other segments, the node potential benefit is used.

It is noteworthy that the **pool** of paths **evolves dynamically** during the search: new paths are added to the pool, others are modified, some may be discarded. Changes in the path set occur due to generation of paths by node deletion or insertion, extraction of subpaths from time-infeasible paths, path optimization by node reordering, path deletion due to pool overflow. Each of these issues is briefly addressed below.

Node deletion Not all requests associated to a node of a given tour are necessarily served by that tour. In particular, after a swap move, some requests previously served may have lost their boarding or alighting node, or, similarly, some of the requests associated to the nodes of the entering path may not find their mate nodes in the paths of the other segments. Such requests are called *orphans* with respect to the current tour. A *profitless* node with respect to a given tour serves only orphan requests: as it yields no profit while increases the cost of the tour, it should be discarded. Clearing paths of profitless nodes in the current tour, possibly originates new paths thus enriching the pool.

Path reduction New paths can also be generated during pool inspection, when searching for the best candidate path for swapping. If the selected path is longer than required by time-feasibility, any subpath extracted from it by deleting optional stops so as to meet time requirements, is a potential candidate for swapping.

Node reordering In a post-optimization-like phase, paths may be modified regarding the order nodes are traversed, to improve the overall efficiency by decreasing path duration. Identifying the optimal node sequence in a segment belongs to the TSP family [11, 9]. Therefore, paths are checked only for local optimality with respect to the 2-opt neighborhood defined by the exchange of a single pair of arcs.

Node insertion Tours are improved by various node deletion or reordering operations resulting in extra time for additional detours. The node insertion or *mending* step attempts to locally improve the solution by inserting additional nodes to increase the number of requests served and, therefore, the total benefit of the tour. Nodes are inserted sequentially, according to their potential benefit evaluation.

Pool management As creation of new paths may cause pool overflow, a pool management strategy is needed, to identify the paths to be deleted. In our case, the total node potential benefit of a path is used as a measure.

6 Computational Results

The goal of the experiments was to verify wether problem DAS1 could be solved efficiently and effectively by the path based meta-heuristic, i.e. in terms of both computing time and solution quality, so that it can be utilized as a building block for more general cases (i.e. several vehicles on several lines).

6.1 Test problem data

Calibration and comparison efforts have been performed on a set of problems derived from actual transit lines. The lines have been selected from the transit networks of one Canadian and one European city, together with representations of the corresponding street networks. Each line has been the seed for several problem instances through variations in the number of segments, optional stops for each segment, the density of requests, the width of time windows, and the time available for each segment.

The actual length of each street represented in the network (through a link) is known. These information have been used for computing distances between all the nodes of the network by way of all pairs shortest path algorithms; here, triangle inequalities hold, but distances are neither euclidean nor symmetric. For each original transit line, a subset of stops (one stop for each segment) has been extracted to act as the compulsory stop set of the flexible line. Each street intersection laying within a maximum distance from any stop of the original transit line is eligible for being an optional stop of the flexible line. Optional stops are then assigned to the respective segment. The network topology of the corresponding instance is thus defined. Finally, costs and times on the arcs of the network have been defined proportional to distances. Beside reflecting a real feature, this allows the algorithm to discriminate among paths serving the same node set in a different order and to privilege the most efficient ones with respect to time. The benefits associated to requests have been defined such that the extra cost of any detour to serve a customer is always compensated by the additional benefit. With such settings, limited time availability is the only constraint which causes requests to be discarded. Time windows at compulsory stops have been set in such a way that extra time is provided for detours within each segment; in fact, for each segment h the maximum duration of a path $(b_{h+1} - a_h)$ has been set equal

to one and a half the duration of the path from compulsory stop f_h to f_{h+1} that was required by the original transit line. Time window widths at compulsory stops vary from 60 to 120 seconds, which is considered a reasonable waiting time for passive users who do not know in advance the arrival time of the vehicle. The number of requests is proportional to the number of segments (25 requests for each segment), and boarding and alighting stops have been randomly chosen in the node set N according to a uniform distribution. Since the problem involves a node selection, with this decision we intended to reproduce the most difficult situation, where there is no substantial difference concerning the number of requests associated to each optional node.

Some basic considerations hold. Instances with more segments are more difficult not only because of the larger size: being time a shared resource between adjacent segments, local decisions on path duration induce chain reactions whose impact depends on the chain length. Other factors kept constant, the difficulty is also related to the width of the geographical area encompassed by each segment. Such factor is controlled by the maximum distance parameter used to select optional stops. Varying these factors several sets of problems have been generated. Three sets have been used to calibrate the various parameters of the meta-heuristic and two others to perform the experiments reported in the next section.

6.2 Testing the Hybrid Meta-heuristic Method

The experimental results reported in the following have been computed on two classes of instances obtained from different lines of the same real transit network of the two cities by the procedure given above; class A refers to instances with 5 segments and maximum distance 0.4 of optional stops; class B is made by instances with 10 segments and maximum distance equal to 0.3; 20 different instances have been generated for each class. The number of optional stops in each instance varies increasing with the number of segments and the value of maximum distance.

In order to assess the quality of the heuristic solutions we exploited the state of the art mathematical tools both to improve the bound as well as to get an integer feasible solution: an arc based model, as mentioned in Section 4, has been implemented and given to CPLEX [6], and run on a parallel workstation for a maximum of 10 hours time each, with a limit of 500.000 nodes and a best bound strategy for the tree exploration. The heuristic and the Lagrangean bound were computed on a Pentium II machine with a 233 MHZ processor and 128 MB memory.

Figure 4 and 5 compare the behavior of the upper bounds; for each instance the value of the Lagrangean bound and the value of the best bound obtained by CPLEX

are reported. Further details are reported in [12]; however, it is worth mentioning that while the computation of the Lagrangean bound requires at most few minutes, CPLEX usually requires a considerable amount of running time before improving on it.

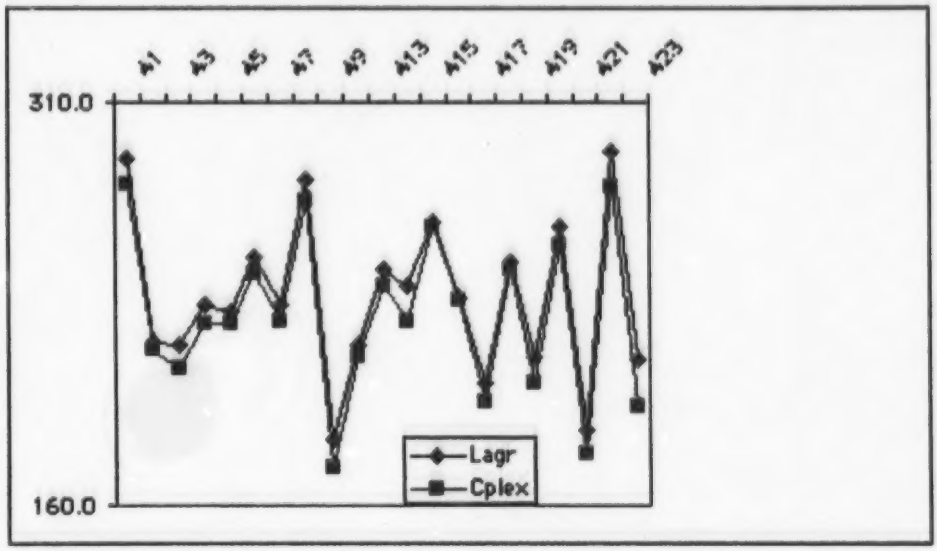

Figure 4: Lagrangean and CPLEX bound for class A

The following tables 6.2 and 6.2 are devoted to the results of the heuristic approach; the solution values obtained after 1, 10 and 60 seconds of running time are reported. Since the procedure includes a number of random parameters, each experiment was repeated ten times for each instance and the average is reported; in order to evaluate the method robustness, the standard deviation is also given.

In order to verify the need for metaheuristics, we also provide the value of the best integer solution obtained by CPLEX with the above mentioned procedure, and the time in seconds needed to compute it.

All solution values are reported in terms of the percentage gap with respect to the best bound known for each instance.

When 0 is reported in the column of CPLEX gap, the optimal integer solution was found. From these data it can be seen that whenever CPLEX is able to solve the problem to optimality, the heuristic procedure is able to provide solutions within the 1.1% quite faster; for more difficult instances, after 100 hours of runnig time the

Instance	1		10		60		cplex	
	σ	gap	σ	gap	σ	gap	gap	time
A1	0.6	0.3	0.0	0.1	0.0	0.1	0.0	145
A2	4.2	13.2	2.0	10.9	1.3	10.4	35.9	1080
A3	4.8	6.0	4.6	3.0	0.5	1.2	2.3	680
A4	4.3	7.7	1.0	4.2	0.7	3.9	39.2	1950
A5	2.5	9.1	1.9	4.7	0.9	3.6	11.1	280
A6	2.6	10.5	2.0	7.9	1.3	7.4	30.4	980
A7	3.5	7.7	3.4	6.5	1.9	5.5	34.1	510
A8	3.4	11.6	2.8	10.0	2.3	8.8	31.9	250
A9	2.0	6.8	1.6	5.2	1.4	4.3	15.2	22640
A12	5.5	8.7	2.8	5.4	1.9	4.1	15.0	910
A13	1.6	7.0	1.5	4.8	1.1	4.4	23.7	520
A14	6.6	6.4	3.2	4.0	2.5	2.6	3.6	790
A15	5.9	6.2	1.2	3.9	1.8	3.5	22.6	4580
A16	3.8	15.0	3.1	13.2	2.1	12.5	42.3	370
A17	3.7	10.8	0.9	8.2	0.0	8.0	24.3	2500
A18	5.1	13.2	3.0	11.8	0.1	11.1	—	10 h
A19	4.6	12.9	3.3	10.4	1.0	8.8	29.9	1990
A20	3.4	6.5	0.5	5.1	0.3	5.0	35.6	440
A21	3.3	11.8	0.7	9.7	0.9	9.4	30.6	830
A22	1.9	1.9	2.7	1.5	3.0	1.1	0.0	611
A23	1.0	0.5	0.1	0.2	0.1	0.2	0.0	1414

Table 1: Gap between the heuristic solutions and the best upper bound: class A

Instance	1		10		60		cplex	
	σ	gap	σ	gap	σ	gap	gap	time
B1	0.3	1.9	0.1	1.8	0.1	1.7	1.6	2140
B2	1.0	5.3	0.6	4.4	0.3	4.3	31.0	2070
B3	1.5	1.1	1.4	1.0	1.3	0.8	0.0	98
B4	0.7	1.1	0.2	0.6	0.0	0.5	0.0	10720
B5	0.6	0.5	0.6	0.4	0.6	0.4	0.0	5469
B6	0.0	2.5	0.0	2.5	0.0	2.5	9.4	7730
B7	0.1	3.4	0.5	3.2	0.6	3.1	27.9	290
B8	1.2	1.4	1.0	1.1	0.7	0.6	0.0	925
B9	0.5	0.6	0.5	0.5	0.4	0.3	0.0	2842
B10	1.4	2.8	0.5	1.9	0.1	1.8	1.9	14590
B11	0.8	1.0	0.6	0.7	0.6	0.7	0.0	414
B12	1.2	2.2	0.6	1.5	0.5	1.2	0.0	855
B13	1.3	3.5	1.7	2.6	1.4	2.0	2.3	870
B14	1.0	1.4	1.1	1.1	1.0	1.1	0.0	200
B15	1.0	2.4	0.9	2.2	0.8	2.1	8.9	100
B16	1.1	5.1	0.8	4.6	0.7	4.6	27.9	150
B17	0.5	1.7	0.6	1.4	0.5	1.2	13.3	15320
B18	1.2	2.3	1.3	0.6	0.5	0.2	0.0	502
B19	1.1	1.2	0.9	0.5	0.9	0.4	0.0	288
B20	0.9	9.5	0.8	8.9	0.6	8.7	58.7	3490

Table 2: Gap between the heuristic solutions and the best upper bound: class B

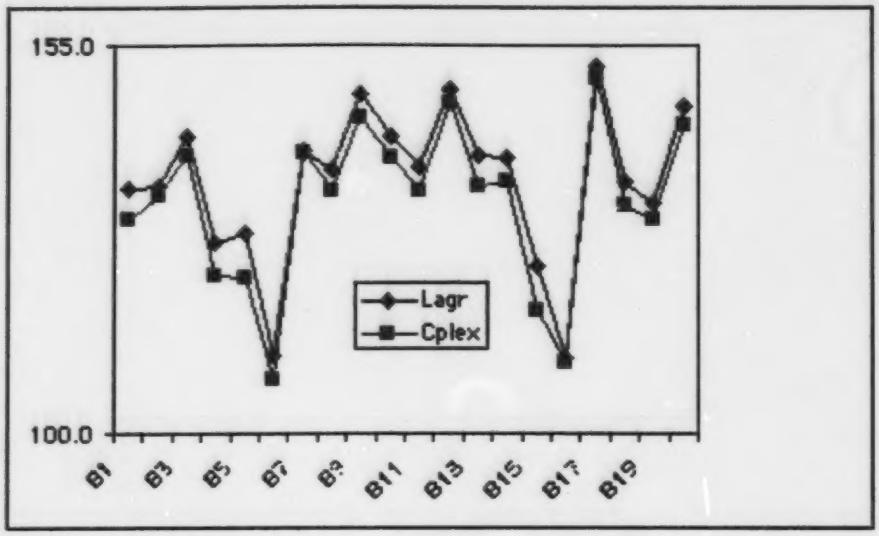

Figure 5: Lagrangean and CPLEX bound for class B

best bound provided by CPLEX is still rather far from the heuristic solution, and in turn, the best integer solution provided by CPLEX is considerably worse than the one provided by the heuristic.

7 Perspectives and Conclusions

In this paper a hybrid metaheuristic method integrating several solution strategies has been proposed for the solution of the basic version of a new flexible transport system. Such a system shares some of the features of traditional fixed lines while offering at the same time a high degree of flexibility. The commercial Mixed Integer Programming solvers are not enough efficient to solve the optimization problem associated to the service management even in the basic case, when requests must be selected for a single tour in a single line. The particular structure of the optimization problem makes even the basic case not suitable to be solved by classical dial-a-ride solution approaches. For these reasons we developed specific algorithmic tools for this problem.

The procedure has been tested on a set of problem instances derived from real life networks, and compared with a state of the art package for integer linear programming. Our method proved quite effective, although for same instances the quality of the solution has not yet been assessed.

Finally, we can state that the basic problem can be efficiently handled by specific methods, which can represent the building block of more general approaches to the multi-vehicle, multi-line case.

8 Acknowledgments

We are grateful to Professor Michael Florian who graciously gave us access to a number of data banks from which a set of realistic test problems could be derived. This project is supported by grants from NSERC Canada, the Fonds FCAR of the Province of Quebec, the Quebec-Italy cooperation program of the Ministry for Foreign Affairs of Italy and the Ministry for International Relations of the Province of Quebec.

References

[1] Borndörfef, R., Gröetschel, M., Klostermeier, F., and Küttner, C. (1997). *Telebus Berlin: Vehicle Scheduling in a Dial-a-Ride System*, Konrad-Zuse Center for Information Systems SC-97-23.

[2] Carraresi, P., Frangioni, A., and Nonato, M. (1996). *Applying Bundle Methods to the Optimization of Polyhedral Functions: An Applications Oriented Development*, **Ricerca Operativa** 25, 5–49.

[3] Crainic, T.G., Guertin, F., Malucelli, F., and Nonato, M. (2000). *Metaheuristics for a class of Demand Responsive Transit Systems*, Publication, Centre de recherche sur les transports, Université de Montréal.

[4] Crainic, T.G., Malucelli, F., and Nonato, M. (2000). *A demand responsive feeder bus system*, Proceedings of the Intelligent Transportation Systems, Torino (Italy).

[5] Crainic, T.G., Malucelli, F., and Nonato, M. (2001). *Many-to-few + few-to-many = an almost personalized transit system* TRISTAN 2001.

[6] *ILOG CPLEX 7.0 Reference Manual*, ILOG Inc., CPLEX Division, 2000.

[7] Desrosiers, J., Dumas, Y., Solomon, M.M., and Soumis, F. (1995). *Time constrained routing and scheduling*, in M.O.Ball et al.(Eds.), *Network Routing*, Handbooks in Operations Research and Management Science 8, Elsevier Science, Amsterdam 35–139.

[8] Ioachim, I, Desrosiers, J., Dumas, Y., Solomon, M.M., and Villeneuve, D. (1995). *A request clustering for door to door handicapped transportation*, **Transportation Science** 29, 63–78.

[9] Laporte, G. (1992). *The Travelling Salesman Problem: An Overview of Exact and Approximate Algorithms*, **European Journal of Operational Research** 59, 231-247.

[10] Cordeau, J.-F., Laporte, G., and Mercier, A. (2000). *A unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows*, Technical Report CRT-2000-03, Centre de Recherche sur les Transports, University of Montreal.

[11] Lawler, E.L., Lenstra, J.K., Rinnoy Kan, A.H.G., and Shmoys, D.B. (1985). *The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley, New York, N.Y.

[12] Malucelli, F., Nonato, M. (2000). *Formulations and bounding procedures for an optimization problems arising in managing a demand adaptive transit system* Working paper.

[13] Malucelli, F., Nonato, M., and Pallottino, S. (1999). *Some proposals on flexible transit*, in T.Ciriani, S. Gliozzi, E.L. Johnson, and R. Tadei (Eds.), *Operations Research in Industry*, Macmillan Press, London, 157–182.

[14] Salvensbergh, M.W.P., and Sol, M. (1995). *The General Pickup and Delivery Problem*, **Transportation Science** 29(1), 17–29.

[15] Shen,Y., Potvin, J.-Y., Rousseau, J.-M., and Roy, S. (1995). *A Computer Assistant for Vehicle Dispatching with Learning Capabilities*, **Annals of Operations Research** 61, 189–211.